Negotiation Games

Javier Esparza and Philipp Hoffmann

Fakultät für Informatik, Technische Universität München, Germany

Abstract. Negotiations, a model of concurrency with multi party negotiation as primitive, have been recently introduced in [**?**,**?**]. We initiate the study of games for this model. We study *coalition problems*: can a given coalition of agents force that a negotiation terminates (resp. block the negotiation so that it goes on forever)?; can the coalition force a given outcome of the negotiation? We show that for arbitrary negotiations the problems are EXPTIME-complete. Then we show that for sound and deterministic or even weakly deterministic negotiations the problems can be solved in PTIME. Notice that the input of the problems is a negotiation, which can be exponentially more compact than its state space.

1 Introduction

In [**?**,**?**], the first author and Jörg Desel have introduced a model of concurrency with multi party negotiation as primitive. The model allows one to describe distributed negotiations obtained by combining "atomic" multi party negotiations, or *atoms*. Each atom has a number of *parties* (the subset of agents involved), and a set of possible outcomes. The parties agree on an outcome, which determines for each party the subset of atoms it is ready to engage in next.

Ill-designed negotiations may deadlock, or may contains useless atoms, i.e., atoms that can never be executed. The problem whether a negotiation is well designed or *sound* was studied in [**?**,**?**]. The main result was the identification of two classes, called deterministic and acyclic weakly deterministic negotiations, for which the soundness problem is tractable: while the problem is PSPACE-complete for arbitrary negotiations, it becomes polynomial for these two classes.

In this paper we start the study of games on negotiations. As for games played on pushdown automata [**?**], vector addition systems with states (VASS) [**?**], counter machines [**?**], or asynchronous automata [**?**], games on negotiations can be translated into games played on the (reachable part of the) state space. However, the number of states of a negotiation may grow exponentially in the number of agents, and so the state space can be exponentially larger than the negotiation. We explore the complexity of solving games *in the size of the negotiation*, not on the size of the state space. In particular, we are interested in finding negotiation classes for which the winner can be decided in polynomial time, thus solving the state space explosion problem.

We study games formalizing the two most interesting questions related to a negotiation. First, can a given *coalition* (i.e., a given subset of agents) force termination of the negotiation? (Negotiations may contain cycles.) Second, can the coalition force a given final outcome?

Our first results show that these two problems are EXPTIME-complete in the size of the negotiation. This is the case even if the negotiation is deterministic, and so it seems as if the tractability results of [?,?] cannot be extended to games. But then, we are able to show that, very surprisingly, the problems are polynomial for deterministic (or even weakly deterministic) negotiations *that are sound*. This is very satisfactory: since unsound negotiations are ill-designed, we are not interested in them anyway. And, very unexpectedly, the restriction to sound negotiations has as collateral effect a dramatic improvement in the complexity of the problem. Moreover, the restriction comes "at no cost", because deciding soundness of deterministic negotiations is also decidable in polynomial time.

Related work. Our games can be seen as special cases of concurrent games [?,?] in which the arena is succinctly represented as a negotiation. Explicit construction of the arena and application of the algorithms of [?,?] yields an exponential algorithm, while we provide a polynomial one.

Negotiations have the same expressive power as 1-safe Petri nets or 1-safe VASS, although they can be exponentially more compact (see [?,?]). Games for unrestricted VASS have been studied in [?]. However, in [?] the emphasis is on VASS with an infinite state space, while we concentrate in the 1-safe case.

The papers closer to ours are those studying games on asynchronous automata (see e.g. [?,?,?]). Like negotiations, asynchronous automata are a model of distributed computation with a finite state space. These papers study algorithms for deciding the existence of distributed strategies for a game, i.e., local strategies for each agent based only on the information the agent has on the global system. Our results identify a special case with much lower complexity than the general one, in which local strategies are even memoryless.

Finally, economists have studied mathematical models of negotiation games, but with different goals and techniques (see e.g. [?]). In our terminology, they typically consider negotiations in which all agents participate in all atomic negotiations. We focus on distributed negotiations, where in particular atomic negotiations involving disjoint sets of agents may occur concurrently.

2 Negotiations: Syntax and Semantics

Negotiations are introduced in [?]. We recall the main definitions. We fix a finite set A of *agents* representing potential parties of negotiations. In [?,?] each agent has an associated set of internal states. For the purpose of this paper the internal states are irrelevant, and so we omit them.

Atoms. A *negotiation atom*, or just an *atom*, is a pair $n = (P_n, R_n)$, where $P_n \subseteq A$ is a nonempty set of *parties*, and R_n is a finite, nonempty set of *outcomes*.

(Distributed) Negotiations. A distributed negotiation is a set of atoms together with a *transition function* $\mathcal{X}$ that assigns to every triple (n, a, r) consisting of an atom n, a party a of n, and an outcome r of n a set $\mathcal{X}(n, a, r)$ of atoms. Intuitively, this is the set of atomic negotiations agent a is ready to engage in after the atom n, if the outcome of n is r.

Formally, given a finite set of atoms N, let $T(N)$ denote the set of triples (n, a, r) such that $n \in N$, $a \in P_n$, and $r \in R_n$. A *negotiation* is a tuple $\mathcal{N} = (N, n_0, n_f, \mathcal{X})$, where $n_0, n_f \in N$ are the *initial* and *final* atoms, and $\mathcal{X} \colon T(N) \to 2^N$ is the *transition function*. Further, $\mathcal{N}$ satisfies the following properties: (1) every agent of A participates in both n_0 and n_f; (2) for every $(n, a, r) \in T(N)$: $\mathcal{X}(n, a, r) = \emptyset$ iff $n = n_f$.

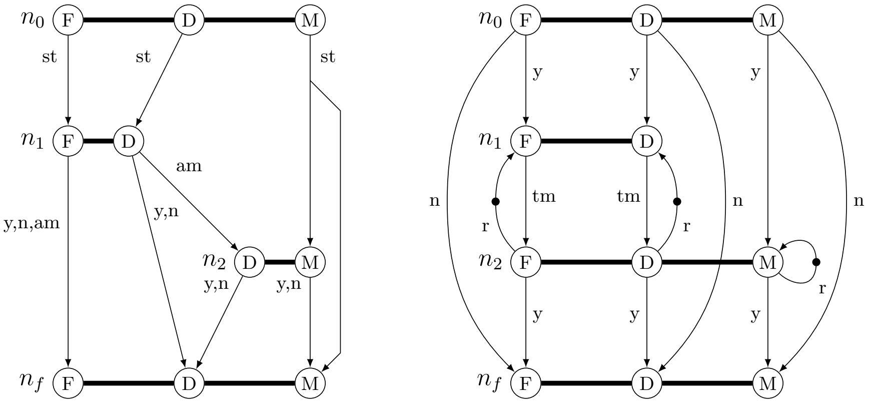

Fig. 1. An acyclic and a cyclic negotiation.

Graphical representation. Negotiations are graphically represented as shown in Figure **??**. For each atom $n \in N$ we draw a black bar; for each party a of P_n we draw a white circle on the bar, called a *port*. For each $(n, a, r) \in T(N)$, we draw a hyper-arc leading from the port of a in n to all the ports of a in the atoms of $\mathcal{X}(n, a, r)$, and label it by r. Figure **??** shows two Father-Daughter-Mother negotiations. On the left, Daughter and Father negotiate with possible outcomes yes (y), no (n), and ask_mother (am). If the outcome is the latter, then Daughter and Mother negotiate with outcomes yes, no. In the negotiation on the right, Father, Daughter and Mother negotiate with outcomes yes and no. If the outcome is yes, then Father and Daughter negotiate a return time (atom n_1) and propose it to Mother (atom n_2). If Mother approves (outcome yes), then the negotiation terminates, otherwise (outcome r) Daughter and Father renegotiate the return time.

Semantics. A *marking* of a negotiation $\mathcal{N} = (N, n_0, n_f, \mathcal{X})$ is a mapping $\boldsymbol{x} \colon A \to 2^N$. Intuitively, $\boldsymbol{x}(a)$ is the set of atoms that agent a is currently ready to engage in next. The *initial* and *final* markings, denoted by $\boldsymbol{x}_0$ and $\boldsymbol{x}_f$ respectively, are given by $\boldsymbol{x}_0(a) = \{n_0\}$ and $\boldsymbol{x}_f(a) = \emptyset$ for every $a \in A$.

A marking $\boldsymbol{x}$ *enables* an atom n if $n \in \boldsymbol{x}(a)$ for every $a \in P_n$, i.e., if every party of n is currently ready to engage in it. If $\boldsymbol{x}$ enables n, then n can take place and its parties agree on an outcome r; we say that (n, r) *occurs.* Abusing language, we will call this pair also an outcome. The occurrence of (n, r) produces a next marking $\boldsymbol{x}'$ given by $\boldsymbol{x}'(a) = \mathcal{X}(n, a, r)$ for every $a \in P_n$, and $\boldsymbol{x}'(a) = \boldsymbol{x}(a)$ for every $a \in A \setminus P_n$. We write $\boldsymbol{x} \xrightarrow{(n,r)} \boldsymbol{x}'$ to denote this.

By this definition, $\boldsymbol{x}(a)$ is always either $\{n_0\}$ or equals $\mathcal{X}(n, a, r)$ for some atom n and outcome r. The marking $\boldsymbol{x}_f$ can only be reached by the occurrence of (n_f, r) (r being a possible outcome of n_f), and it does not enable any atom. Any other marking that does not enable any atom is a *deadlock.*

Reachable markings are graphically represented by placing tokens (black dots) on the forking points of the hyper-arcs (or in the middle of an arc). Figure **??** shows on the right a marking in which F and D are ready to engaging n_1 and M is ready to engage in n_2.

We write $\boldsymbol{x}_1 \xrightarrow{\sigma}$ to denote that there is a sequence

$$\boldsymbol{x}_1 \xrightarrow{(n_1,r_1)} \boldsymbol{x}_2 \xrightarrow{(n_2,r_2)} \cdots \xrightarrow{(n_{k-1},r_{k-1})} \boldsymbol{x}_k \xrightarrow{(n_k,r_k)} \boldsymbol{x}_{k+1} \cdots$$

such that $\sigma = (n_1, r_1) \ldots (n_k, r_k) \ldots$. If $\boldsymbol{x}_1 \xrightarrow{\sigma}$, then σ is an *occurrence sequence* from the marking $\boldsymbol{x}_1$, and $\boldsymbol{x}_1$ enables σ. If σ is finite, then we write $\boldsymbol{x}_1 \xrightarrow{\sigma} \boldsymbol{x}_{k+1}$ and say that $\boldsymbol{x}_{k+1}$ is *reachable* from $\boldsymbol{x}_1$.

Soundness. A negotiation is *sound* if *(a)* every atom is enabled at some reachable marking, and *(b)* every occurrence sequence from the initial marking either leads to the final marking $\boldsymbol{x}_f$, or can be extended to an occurrence sequence that leads to $\boldsymbol{x}_f$.

The negotiations of Figure **??** are sound. However, if we set in the left negotiation $\mathcal{X}(n_0, \mathtt{M}, \mathtt{st}) = \{n_2\}$ instead of $\mathcal{X}(n_0, \mathtt{M}, \mathtt{st}) = \{n_2, n_f\}$, then the occurrence sequence $(n_0, \mathtt{st})(n_1, \mathtt{yes})$ leads to a deadlock.

Determinism and weak determinism. An agent $a \in A$ is *deterministic* if for every $(n, a, r) \in T(N)$ such that $n \neq n_f$ there exists an atom n' such that $\mathcal{X}(n, a, r) = \{n'\}$.

The negotiation $\mathcal{N}$ is *weakly deterministic* if for every $(n, a, r) \in T(N)$ there is a deterministic agent b that is a party of every atom in $\mathcal{X}(n, a, r)$, i.e., $b \in P_{n'}$ for every $n' \in \mathcal{X}(n, a, r)$. In particular, every reachable atom has a deterministic party. It is *deterministic* if all its agents are deterministic.

Graphically, an agent a is deterministic if no proper hyper-arc leaves any port of a, and a negotiation is deterministic if there are no proper hyper-arcs. The negotiation on the left of Figure **??** is not deterministic (it contains a proper hyper-arc for Mother), while the one on the right is deterministic.

3 Negotiation Games

We study a setting that includes, as a special case, the questions about coalitions mentioned in the introduction: Can a given coalition (subset of agents) force termination of the negotiation? Can the coalition force a given final outcome?

We need the notion of an independent set of atoms.

Definition 1. *A set of atoms S is independent if no two distinct atoms of S share an agent, i.e., $R_n \cap R_{n'} = \emptyset$ for every $n, n' \in S$, $n \neq n'$.*

It follows immediately from the semantics that if a marking $\boldsymbol{x}$ enables all atoms of S, then there is a unique marking $\boldsymbol{x}'$ such that $\boldsymbol{x} \xrightarrow{\sigma} \boldsymbol{x}'$ for every sequence $\sigma = (n_1, r_1) \dots (n_k, r_k)$ such that each atom of S appears exactly once in σ. In other words, $\boldsymbol{x}'$ depends only on the outcomes of the atoms, and not on the order in which they are fired.

A *negotiation arena* is a negotiation whose set N of atoms is partitioned into two sets N_1 and N_2. We consider concurrent games [**?**,**?**] with three players called Player 1, Player 2, and Scheduler. At each step, Scheduler chooses a nonempty set of independent atoms among the atoms enabled at the current marking of the negotiation arena. Then, Player 1 and Player 2, independently of each other, select an outcome for each atom in $S \cap N_1$ and $S \cap N_2$, respectively. Finally, these outcomes are fired in any order, and the game moves to the unique marking $\boldsymbol{x}'$ mentioned above. The game terminates if it reaches a marking enabling no atoms, otherwise it continues forever.

Formally, a partial play is a sequence of tuples $(S_i, F_{i,1}, F_{i,2})$ where each $S_i \subseteq N$ is a set of independent atoms and $F_{i,j}$ assigns to every $n \in S_i \cap N_j$ an outcome $r \in R_n$. Furthermore it must hold that every atom $n \in S_i$ is enabled after all atoms in $S_0, ..., S_{i-1}$ have occurred with the outcomes specified by $F_{0,1}, F_{0,2}, ..., F_{i-1,1}, F_{i-1,2}$. A play is a partial play that is either infinite or reaches a marking enabling no atoms. For a play π we denote by π_i the partial play consisting of the first i tuples of π.

We consider two different winning conditions. In the *termination game*, Player 1 wins a play if the play ends with n_f being fired (notice that no atom is enabled after n_f fires); otherwise Player 2 wins. In the *final-outcome game*, we select for each agent a a set of outcomes G_a such that $n_f \in \mathcal{X}(n, a, r)$ for $r \in G_a$ (that is, after any outcome $r \in G_a$, agent a is ready to terminate). Player 1 wins if the the play ends with n_f being fired, and for each agent a the last outcome (n, r) of the play such that a is a party of n belongs to G_a.

A strategy σ for Player $j, j \in \{1, 2\}$ is a partial function that, given a partial play $\pi = (S_0, F_{0,1}, F_{0,2}), .., (S_i, F_{i,1}, F_{i,2})$ and a set of atoms S_{i+1} returns a function $F_{i+1,j}$ according to the constraints above. A play π is said to be played according to a strategy σ of Player j if for all i, $\sigma(\pi_i, S_{i+1}) = F_{i+1,j}$. A strategy σ is a winning strategy for Player j if he wins every play that is played according to σ. Player j is said to win the game if he has a winning strategy. Notice that if Player 1 has a winning strategy then it wins every play against any pair of strategies for Player 2 and Scheduler.

Definition 2. *Let $\mathcal{N}$ be a negotiation arena. The termination (resp. final-outcome) problem for $\mathcal{N}$ consists of deciding whether Player 1 has a winning strategy for the termination game (final-outcome game).*

Consider the example negotiation in figure **??** with Father (F), Mother (M) and two daughters (D$_1$, D$_2$) as agents. The solid edges for the daughters from n_2 to n_3, n_4 to n_6, and n_5 to n_7 are “ask the other parent” outcomes, while

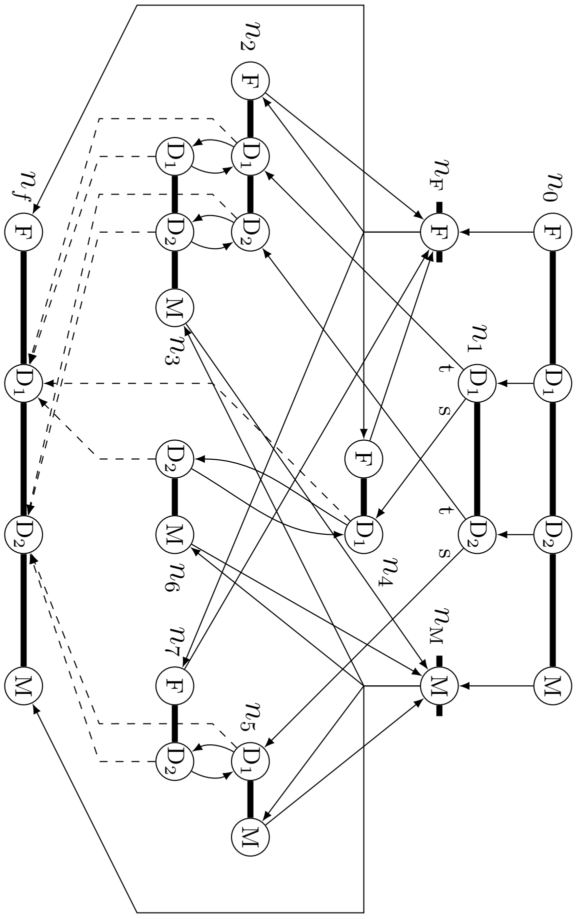

Fig. 2. A negotiation of Father, Mother and two daughters

the dashed edges represent the "yes" and "no" outcomes. Assume the daughters form a coalition. Then $N_1 = \{n_1, n_2, n_3\}$, since these are the atoms where the daughters have a majority; N_2 contains all other atoms. Can the daughters force termination?

At atom n_1 the daughters have the choice to split up and talk to one parent each (outcome `s`) or talk to their father together (outcome `t`). If the daughters choose outcome `s`, then Father and Mother can force an infinite loop, either between n_4 and n_6, or between n_5 and n_7. On the contrary, if the daughters choose to stay together, then, since they own atom n_2, they can force a "yes" or "no" outcome, and therefor termination.

The questions whether a coalition $\mathcal{C}$ of agents can force termination or a certain outcome are special instances of the termination and final-outcome problems. In these instances, an atom n belongs to N_1—the set of atoms controlled by Player 1— iff a strict majority of the agents of n are members of $\mathcal{C}$.

3.1 Coalitions

Before we turn to the termination and final-outcome problems, we briefly study coalitions. Intuitively, a coalition controls all the atoms where it has strict majority. We show that while the definition of the partition of the atoms N according to the participating agents may seem restricting, this is not the case: In all cases but the deterministic sound case, any partition can be reached, possibly by adding agents.

We define the partition of N via a partition of the agents: Let the agents A be partitioned into two sets A_1 and A_2. Define $N_1 = \{n \in N : |P_n \cap A_1| > |P_n \cap A_2|\}$, $N_2 = N \backslash N_1$. Note that ties are controlled by A_2.

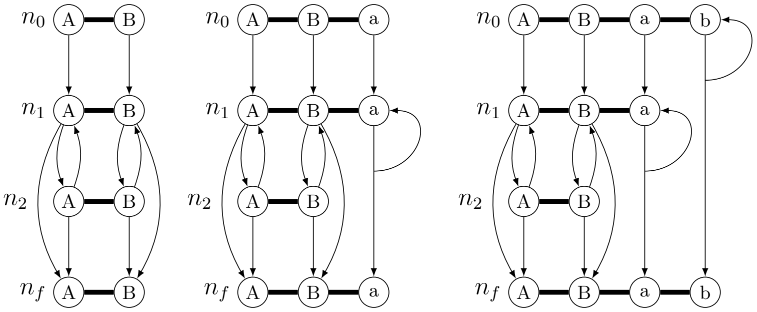

Fig. 3. Atom control via additional agents

We first show that in the nondeterministic and weakly deterministic case, this definition is equivalent to one where we decide control for each atom and not for each agent: Consider the example given in Figure **??**. On the left a deterministic negotiation with two agents is given. Assume the coalitions are

$A_1 = \{A\}$ and $A_2 = \{B\}$. By the definition above, $N_2 = N$, thus coalition A_2 controls every atom. We want to change control of n_1 so that A_1 controls n_1, changing the negotiation to a weakly deterministic one on the way. We add an additional agent a that participates in n_0, n_1, n_f as shown in Figure **??** in the middle and set $A_1 = \{A, a\}$. Now A_1 controls n_1 but also n_0 and n_f. We therefore add another agent b that participates in n_0, n_f as shown in Figure **??** on the right. Now the partition of atoms is exactly $N_1 = \{n_1\}$ and $N_2 = \{n_0, n_2, n_f\}$, as desired. In general, by adding nondeterministic agents to the negotiation, we can change the control for each atom individually. For each atom n whose control we wish to change, we add a number of agents to that atom, the initial atom n_0 and final atom n_f. We add nondeterministic edges for these agent from n_0 to $\{n, n_f\}$ for each outcome of n_0 and from n to $\{n, n_f\}$ for each outcome of n. It may be necessary to add more agents to n_0 that move to n_f as to not change the control of n_0 or n_f. This procedure changes the control of n while preserving soundness and weak determinism.

We proceed by showing that in the deterministic case, we cannot generate any atom control by adding more agents.

Lemma 1. *We cannot add deterministic agents to the negotiation above in a manner, such that soundness is preserved and Player 1 controls n_1, Player 2 controls n_2.*

Proof. Consider again the deterministic negotiation game on the left of Figure **??**. Assume we have added deterministic agents such that Player 1 controls n_1. After the occurring sequence $x_0 \xrightarrow{(n_0,a)} x_1 \xrightarrow{(n_1,a)} x_2$, those additional agents have moved deterministically, either to n_1 or n_f. [1] If any agent remains in n_1, choosing outcome b in n_2 leads to a deadlock, otherwise, choosing a leads to a deadlock. Thus the negotiation is no longer sound. □

4 The termination problem

We turn to the general complexity of the termination problem. To show EXPTIME-membership, we use the results of [**?**]. We begin by introducing the notion of a concurrent game structure.

A concurrent game structure is a tuple $S = (k, Q, \Pi, \pi, d, \delta)$ where

- $k \geq 1$ is a natural number, the number of players.
- Q is a finite set of states.
- Π is a finite set of propositions.
- π assigns every state $q \in Q$ a set of propositions that are true in q.
- d assigns every player $a \in \{1, ..., k\}$ and every state $q \in Q$ a natural number $d_a(q) \geq 1$ of possible moves. We identify these moves with natural numbers $1, ..., d(a, q)$. For each state q, we write $D(q)$ for the set $\{1, ..., d_1(q)\} \times ... \times \{1, ..., d_k(q)\}$ of move vectors.

[1] Moving to n_2 would change the control there, thus additional agents have to be added to n_2, we then can use a similar argument as follows by exchanging the roles of n_1 and n_2.

- δ is the transition function that assigns a state $q \in Q$ and a move vector $(j_1, ..., j_k) \in D(q)$ a state $\delta(q, j_1, ..., j_k)$ that results from state q if every player $a \in \{1, ..., k\}$ chooses move j_a.

The following is one of the results from [**?**]:

Theorem 1. *Model checking for reachability objectives on concurrent game structures possible in time linear in the number of transitions of the game and the length of the reachability formula.*

Using this result, it is easy to see that the termination problem can be solved in exponential time.

Theorem 2. *The termination problem is in EXPTIME.*

Proof. We construct a 3-player concurrent game structure and a reachability objective such that Player 1 wins the negotiation game iff she wins this new game against the coalition of the other two players. The game has single exponential size in the size of the negotiation arena.

The states of the game are either markings $\boldsymbol{x}$ of the negotiation, or pairs $(\boldsymbol{x}, N_{\boldsymbol{x}})$, where $\boldsymbol{x}$ is a marking and $N_{\boldsymbol{x}}$ is an independent set of atoms enabled at $\boldsymbol{x}$. Nodes $\boldsymbol{x}$ belong to Scheduler, who chooses a set $N_{\boldsymbol{x}}$, after which the play moves to $(\boldsymbol{x}, N_{\boldsymbol{x}})$. At nodes $(\boldsymbol{x}, N_{\boldsymbol{x}})$ Players 1 and 2 concurrently select subsets of $N_{\boldsymbol{x}}$, and depending on their choice the play moves to a new marking.

The states of the concurrent game structure are the following:

$$\begin{aligned} Q = \{\boldsymbol{x} : \boldsymbol{x} \text{ a reachable marking}\} \cup \{(\boldsymbol{x}, N_{\boldsymbol{x}}) : \boldsymbol{x} \text{ a reachable marking},\\ N_{\boldsymbol{x}} \text{ a set of independent atoms enabled in } \boldsymbol{x}\} \end{aligned}$$

Only one proposition is needed to mark the final state:

$$\Pi = \{\text{final}\}$$

$$\pi(\boldsymbol{x}) = \begin{cases} \text{final} & \text{if } \boldsymbol{x} = \boldsymbol{x}_f \\ \emptyset & \text{otherwise} \end{cases}$$

$$\pi((\boldsymbol{x}, N_{\boldsymbol{x}})) = \emptyset$$

The move vectors, given by the cartesian product of the possible moves of each player, are defined as follows:

- $D(\boldsymbol{x}) = \{1\} \times \{1\} \times S$ is the set of move vectors from state $\boldsymbol{x}$, where S is the set of all sets of independent atoms enabled in $\boldsymbol{x}$.
- $D((\boldsymbol{x}, N_{\boldsymbol{x}})) = \{F_1\} \times \{F_2\} \times \{1\}$ is the set of move vectors from state $(\boldsymbol{x}, N_{\boldsymbol{x}})$, where F_i is the set of all functions assigning each $n \in N_{\boldsymbol{x}} \cap N_i$ an outcome $r \in R_n$.

The transition function δ is defined by $\delta(\boldsymbol{x}, (1, 1, s)) = (\boldsymbol{x}, s)$, and $\delta((\boldsymbol{x}, N_{\boldsymbol{x}}), (f_1, f_2, 1)) = \boldsymbol{x}'$, where $\boldsymbol{x}'$ is the result of firing from state $\boldsymbol{x}$ all the atoms of $N_{\boldsymbol{x}}$ with the outcomes specified by f_1, f_2. Notice that, since the atoms are independent, $\boldsymbol{x}'$ does not depend on the order in which they are fired.

Player 1 wins if the play reaches the final marking $\boldsymbol{x}_f$, or equivalently, the reachability objective is $\Diamond$final. Using theorem **??**, the result follows. □

Unfortunately, there is a matching lower bound.

Theorem 3. *The termination problem is EXPTIME-hard even for negotiations in which every reachable marking enables at most one atom.*

Proof. The proof is by reduction from the acceptance problem for alternating, linearly-bounded Turing machines (TM). We are given an alternating TM M with transition relation δ, and an input x of length n. We assume that M always halts in one of two designated states q_{accept} or q_{reject}, and does so immediately after reaching one of those states.

We define a negotiation with two agents P and I modeling the head position and internal state of M, and one agent C_k for each cell. The set of atoms contains one atom $n_{q,\alpha,k}$ for each triple (q, α, k), where q is a control state of M, k is the current position of the head, and α is the current letter in the k-th tape cell.

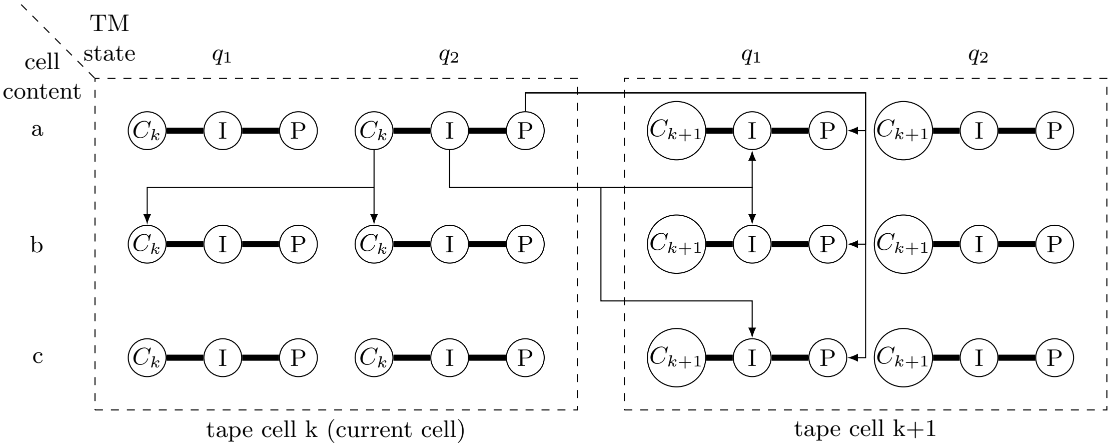

Fig. 4. Part of the negotiation representing a TM, the edges drawn represent the transition $(q_2, a) \rightarrow (q_1, b, R)$

Figure **??** shows a part of the negotiation corresponding to a TM with two internal states, q_1, q_2 and three tape symbols a, b, c. The part shown represents two cells of the tape as well as the movement of the agents corresponding to the TM taking the transition $(q_2, a) \rightarrow (q_1, b, R)$.

The parties of the atom $n_{q,\alpha,k}$ are I, P and C_k, in particular, I and P are agents of all atoms. The atom $n_{q,\alpha,k}$ has one outcome r_τ for each element $\tau \in \delta(q,\alpha)$, where τ is a triple consisting of a new state, a new tape symbol, and a direction for the head. We informally define the function $\mathcal{X}(n_{q,\alpha,k}, C_k, r_\tau)$ by means of an example. Assume that, for instance, $\tau = (q', \beta, R)$, i.e., at control state q and with the head reading α, the machine can go to control state q', write β, and move the head to the right. Then we have: (i) $\mathcal{X}(n_{q,\alpha,k}, I, r_\tau)$ contains all atoms of the form $n_{q',\_,\_}$ (where $\_$ stands for a wild card); (ii) $\mathcal{X}(n_{q,\alpha,k}, P, r)$ contains the atoms $n_{\_,\_,k+1}$; and (iii) $\mathcal{X}(n_{q,\alpha,k}, C_k, r_\tau)$ contains all atoms $n_{\_,\beta,\_}$. If atom $n_{q,\alpha,k}$ is the only one enabled and the outcome r_τ fires, then clearly in the new marking the only atom enabled is $n_{q',\beta,k+1}$. So every reachable marking enables at most one atom.

Finally, the negotiation also has an initial atom that, loosely speaking, takes care of modeling the initial configuration.

The partition of the atoms is: an atom $n_{q,\alpha,k}$ belongs to N_1 if q is an existential state of M, and to N_2 if it is universal. It is easy to see that M accepts x iff Player 1 has a winning strategy.

Notice that if no reachable marking enables two or more atoms, Scheduler never has any choice. Therefore, the termination problem is EXPSPACE-hard even if the strategy for Scheduler is fixed.

Formally, we are given an alternating TM $M = (Q, \mathit{\Gamma}, \delta, q_0, g)$ with tape alphabet $\mathit{\Gamma}$ and a function $g \colon Q \to \{\wedge, \vee, \text{accept}, \text{reject}\}$ that determines the state type, and an input x of length n. We assume that M always halts, does so in designated states q_{accept} or q_{reject}, and does so immediately when reaching one of those. We define a negotiation game $\mathcal{N}_{M,x}$ as follows:

- The set of agents is

$$A = \{I, P\} \cup \{C_k : k \in \{0, ..., n-1\}\}$$

- The set of atoms is

$$N = \{n_0, n_f\} \cup \{n_{q,\alpha,k} : (q, \alpha, k) \in Q \times \mathit{\Gamma} \times \{0, ..., n-1\}\}$$

- The set N_1 of atoms owned by Player 1 contains n_0, n_f, and all atoms $n_{q,\alpha,k}$ such that $g(q) \neq \wedge$. The set N_2 contains all other atoms.
- The set of parties of each atom is given by $P_{n_0} = P_{n_f} = A$, and $P_{n_{q,\alpha,k}} = \{I, P, C_k\}$ for every $n_{q,\alpha,k} \in N$.
- The set of outcomes of each atom is defined as follows: $R_{n_0} = \{\text{st}\}$, $R_{n_f} = \{\text{end}\}$, and for every $n_{q,\alpha,k} \in N$

$$R_{n_{q,\alpha,k}} = \begin{cases} \{\text{f}\} & \text{if } q = q_{accept} \\ \{\text{f}\} & \text{if } q = q_{reject} \\ \{r_\tau : \tau \in \delta(q,\alpha)\} & \text{otherwise} \end{cases}$$

- The $\mathcal{X}(\_, I, \_)$ function for agent I is given by:

$$\begin{aligned}
\mathcal{X}(n_0, I, \mathrm{st}) &= \{n_{q_0,\alpha,0} : \alpha \in \Gamma\} \\
\mathcal{X}(n_{q,\alpha,k}, I, r_\tau) &= \begin{cases} \{n_{q',\alpha,k-1} : \alpha \in \Gamma\} & \text{if } \tau = (q', \_, L) \\ \{n_{q',\alpha,k+1} : \alpha \in \Gamma\} & \text{if } \tau = (q', \_, R) \end{cases} \\
\mathcal{X}(n_{q_{accept},\alpha,k}, I, \mathrm{f}) &= \{n_f\} \\
\mathcal{X}(n_{q_{reject},\alpha,k}, I, \mathrm{s}) &= \{n_{q_{reject},\alpha,k}\}
\end{aligned}$$

- The $\mathcal{X}(\_, P, \_)$ function for agent P is given by $\mathcal{X}(n, P, r) = \mathcal{X}(n, I, r)$ for all $n \in N, r \in P_n$.
- Finally, the $\mathcal{X}(\_, C_k, \_)$ function for agent C_k (the only part of the definition that depends on the input x to M) is given by

$$\begin{aligned}
\mathcal{X}(n_0, C_k, \mathrm{st}) &= \{n_{q,\alpha,k} : q \in Q, \alpha \text{ the initial value of cell } k\} \cup \{n_f\} \\
\mathcal{X}(n_{q,\alpha,k}, C_k, r_\tau) &= \{n_f\} \cup \{n_{q',\beta,k} : q' \in Q\} \text{ if } \tau = (\_, \beta, \_) \\
\mathcal{X}(n_{q_{accept},\alpha,k}, C_k, \mathrm{f}) &= \{n_f\} \\
\mathcal{X}(n_{q_{reject},\alpha,k}, C_k, \mathrm{s}) &= \{n_{q_{reject},\alpha,k}\}
\end{aligned}$$

Observe that, in general, the negotiation $\mathcal{N}_{M,x}$ is nondeterministic and unsound. Initially, we have $\boldsymbol{x}_0(a) = \{n_0\}$ for every agent a, and so only the atom n_0 is enabled. Thereafter, at any point in time, exactly one atom $n_{q,\alpha,k}$ is enabled until $n_{q_{accept},\alpha,k}$ or $n_{q_{reject},\alpha,k}$ is reached. Finally, n_f is enabled iff $n_{q_{accept},\alpha,k}$ was reached. Since the outcome of $n_{q,\alpha,k}$ is chosen by Player 1 when q is an existential state, and by Player 2 when it is a universal state, Player 1 has a winning strategy iff M accepts x. □

Notice that if no reachable marking enables two or more atoms, Scheduler never has any choice. Therefore, the termination problem is EXPTIME-hard even if the strategy for Scheduler is fixed.

A look at points (i)-(iii) in the proof of this theorem shows that the negotiations obtained by the reduction are highly nondeterministic. In principle we could expect a better complexity in the deterministic case. However, this is not so.

Theorem 4. *The termination problem is EXPTIME-hard even for deterministic negotiations in which every reachable marking enables at most one atom.*

Proof. Sketch. (See the appendix for details.) We modify the construction of Theorem **??** so that it yields a deterministic negotiation. The old construction has an atom $n_{q,\alpha,k}$ for each state q, tape symbol α, and cell index k, with I, P, and C_k as parties.

The new construction adds atoms $n_{q,k}$, with I and P as parties, and $n_{\alpha,k}$, with P and C_k as parties. Atoms $n_{q,k}$ have an outcome for each tape symbol α, and atoms $n_{\alpha,k}$ have an outcome for each state q. New atoms are controlled by Player 1.

In the new construction, after firing the outcome of $n_{q,\alpha,k}$ for transition $(q', \beta, R) \in \delta(q, \alpha)$, agent C_k moves to $n_{\beta,k}$, and agents I and P move to $n_{q',k+1}$.

Intuitively, C_k waits for the head to return to cell k, while agents I and P proceed. Atom $n_{q',k+1}$ has an outcome for every tape symbol γ. Intuitively, at this atom Player 1 guesses the current tape symbol in cell $k+1$; the winning strategy corresponds to guessing right. After guessing, say, symbol γ, agent I moves directly to $n_{q',\gamma,k+1}$, while P moves to $n_{\gamma,k+1}$. Atom $n_{\gamma,k+1}$ has one outcome for every state of M. Intuitively, Player 1 now guesses the current control state q', after which both P and C_{k+1} move to $n_{q',\gamma,k+1}$. Notice that all moves are now deterministic.

If Player 1 follows the winning strategy, then the plays mimic those of the old construction: a step like $(n_{q,\alpha,k}, (q', \beta, R))$ in the old construction, played when the current symbol in cell $k+1$ is γ, is mimicked by a sequence $(n_{q,\alpha,k}, (q', \beta, R))$ $(n_{q',k+1}, \gamma)$ $(n_{\gamma,k+1}, q')$ of moves in the new construction. □

5 Termination in sound deterministic negotiations

In [**?**,**?**] we showed that the soundness problem (deciding whether a negotiation is sound) can be solved in polynomial time for deterministic negotiations and acyclic, weakly deterministic negotiations (the case of cyclic weakly deterministic negotiations is open), while the problem is PSPACE-complete for arbitrary negotiations. Apparently, Theorem **??** proves that the tractability of deterministic negotiations stops at game problems. We show that this is not the case. Well-designed negotiations are sound, since otherwise they contain atoms that can never occur (and can therefore be removed), or they can reach a deadlock. Therefore, we are only interested in the termination problem for sound negotiations. We prove that for sound deterministic negotiations (in fact, even for the larger class of weakly deterministic negotiations) the termination and final-outcome problems are polynomial. For this, we show that the well-known attractor construction for reachability games played on graphs as arenas can be "lifted" to sound and weakly deterministic negotiation arenas.

Definition 3. *Let $\mathcal{N}$ be a negotiation arena with a set of atoms $N = N_1 \cup N_2$. Given $n \in N$, let $P_{n,det}$ be the set of deterministic agents participating in n.*

The attractor of the final atom n_f is $\mathcal{A} = \bigcup_{k=0}^{\infty} \mathcal{A}_k$, where $\mathcal{A}_0 = \{n_f\}$ and

$$\begin{aligned}\mathcal{A}_{k+1} = \mathcal{A}_k \cup & \{n \in N_1 : \exists r \in R_n \forall a \in P_{n,det} : \ \mathcal{X}(n,a,r) \in \mathcal{A}_k\} \\ \cup & \{n \in N_2 : \forall r \in R_n \forall a \in P_{n,det} : \ \mathcal{X}(n,a,r) \in \mathcal{A}_k\}\end{aligned}$$

Given a marking $\boldsymbol{x} \neq \boldsymbol{x}_f$ of $\mathcal{N}$ and a deterministic agent a, the attractor position of a at $\boldsymbol{x}$ is the smallest k such that $\boldsymbol{x}(a) \in \mathcal{A}_k$, or ∞ if $\boldsymbol{x}(a) \notin \mathcal{A}$. Let $a_1, \ldots, a_k$ be the deterministic agents of $\mathcal{N}$. The position vector of $\boldsymbol{x}$ is the tuple $(p_1, ..., p_k)$ where we assume that $a_1, \ldots, a_k$ are the deterministic agents of $\mathcal{N}$, and p_i is the attractor position of a_i.

Theorem 5. *Let $\mathcal{N}$ be a sound, weakly deterministic negotiation arena. Player 1 has a winning strategy in the termination game iff $n_0 \in \mathcal{A}$.*

Proof. We start with an observation: If all deterministic agents are ready to take part in n_f, then n_f and only n_f is enabled. Indeed, since deterministic agent are only ready to engage in at most one atom, the only atom with a deterministic party that can be enabled is n_f. Moreover, by weak determinism every atom has a deterministic party, and so no atom other than n_f can be enabled. Finally, by soundness at least one atom is enabled, and so n_f is the only enabled atom.

($\Leftarrow$): Assume that $n_0 \in \mathcal{A}$. We fix the *attractor strategy* for Player 1. We define the attractor index of an atom $n \in \mathcal{A}$ as the smallest k such that $n \in \mathcal{A}_k$. The strategy for an atom $n \in N_1 \cap \mathcal{A}$ is to choose any outcome such that all deterministic parties of n move to an atom of smaller attractor index; formally, we choose any outcome r such that for every deterministic party a the singular atom in $\mathcal{X}(n, a, r)$ has smaller attractor index than n. Such an outcome exists by construction of $\mathcal{A}$. For atoms $n \in N_1 \setminus \mathcal{A}$ we choose an arbitrary atom. Notice that this strategy is not only memoryless, but also independent of the current marking.

We show that the attractor strategy is winning. By the definition of the game, we have to prove that every play following the strategy ends with n_f being fired. By the observation above, it suffices to prove that the play reaches a marking at which every deterministic agent is ready to engage in n_f.

Assume there is a play π where Player 1 plays according to the attractor strategy, which never reaches such a marking. Then the play never reaches the final marking $\boldsymbol{x}_f$ either. We claim that at all markings reached along π, the deterministic agents are only ready to engage in atoms of $\mathcal{A}$. We first observe that, initially, all deterministic agents are ready to engage in n_0, and $n_0 \in \mathcal{A}$. Now, assume that in some marking reached along π the deterministic agents are only ready to engage in atoms of $\mathcal{A}$. Then, by weak determinism, all enabled atoms belong to $\mathcal{A}$, and therefore also the atoms chosen by Scheduler. By the definition of $\mathcal{A}$, after firing atoms of $N_2 \cap \mathcal{A}$ the deterministic agents are ready to engage in atoms of $\mathcal{A}$ only; by the definition of the attractor strategy, the same holds atoms of $N_1 \cap \mathcal{A}$. This concludes the proof of the claim.

Since all markings $\boldsymbol{x}$ reached along π satisfy $\boldsymbol{x} \neq \boldsymbol{x}_f$ and $\boldsymbol{x}(a) \in \mathcal{A}$ for every deterministic agent a, they all have an associated attractor position vector whose components are natural numbers. Let P_k denote the position vector of the marking reached after $k \geq 0$ steps in π. Initially only the initial atom n_0 is enabled, and so $P_0 = (k_0, k_0, \ldots, k_0)$, where k_0 is the attractor position of n_0. We have $k_0 < |N|$, the number of atoms of the negotiation arena $\mathcal{N}$. Given two position vectors $P = (p_1, ..., p_k)$, $P' = (p'_1, ..., p'_k)$, we say $P \prec P'$ if $p_i \leq p'_i$ for every $1 \leq i \leq k$, and $p_i < p'_i$ for at least one $1 \leq i \leq k$. By the definition of the attractor strategy, the sequence $P_0, P_1, ...$ of attractor positions satisfies $P_0 \succ P_1 \succ P_2 \ldots$. Since $\prec$ is a well-founded order, the sequence is finite, i.e., the game terminates. By the definition of the game, it terminates at a marking that does not enable any atom. Since, by assumption, the play never reaches the final marking $\boldsymbol{x}_f$, this marking is a deadlock, which contradicts the soundness of $\mathcal{N}$.

($\Rightarrow$): Let $\mathcal{B} = N \setminus \mathcal{A}$, and assume $n_0 \in \mathcal{B}$. We give a winning strategy for Player 2. The strategy for an atom $n \in N_2 \cap \mathcal{B}$ is to choose any outcome r such that at least one deterministic agent moves to an atom not in $\mathcal{A}$, i.e., such that $\mathcal{X}(n, a, r) \notin \mathcal{A}$ for at least one deterministic agent a. Such an outcome exists because, since $n \in \mathcal{B}$, we have $n \notin \mathcal{A}_k$ for every k, and so by definition and monotonicity of $\mathcal{A}_k$ there exists $r \in R_n$ such that $\mathcal{X}(n, a, r) \notin \mathcal{A}_k$ for some deterministic agent a and every k. For atoms in $N_2 \setminus \mathcal{B}$ we chose an arbitrary outcome.

We show that this strategy is winning for Player 2. Once again, because the negotiation is sound, no play played according to this strategy ends in a deadlock. So we have to prove that every game played following the strategy never ends. By the observation above, it suffices to prove that the play never reaches a marking at which every deterministic agent is ready to engage in n_f. Further, since $n_f \in \mathcal{A}$, it suffices to show that for every marking $\boldsymbol{x}$ reached along the play there is a deterministic agent a such that $\boldsymbol{x}(a) \in \mathcal{B}$.

Initially all deterministic agents are only ready to engage in n_0, and $n_0 \in \mathcal{B}$. Now, assume that at some marking $\boldsymbol{x}$ reached along π there is a deterministic agent a that satisfies $\boldsymbol{x}(a) = n$ and $n \in \mathcal{B}$. We prove that the same holds for the marking $\boldsymbol{x}'$ reached after one step of the play. If n is not enabled at $\boldsymbol{x}$, then we have $\boldsymbol{x}'(a) = \boldsymbol{x}(a) \in \mathcal{B}$, and we are done. The same holds if n is enabled at $\boldsymbol{x}$, but is not selected by the Scheduler. If n is enabled and selected by the Scheduler, there are two possible cases. If $n \in N_1$ then by the definition of $\mathcal{A}$ for every outcome r of n there is a deterministic agent a such that $\mathcal{X}(n_0, a, r) \in \mathcal{B}$, and so $\boldsymbol{x}'(a) \in \mathcal{B}$. If $n \in N_2$, then by definition the strategy chooses an outcome r for which there is an agent a such that $\mathcal{X}(n_0, a, r) \in \mathcal{B}$, and we again get $\boldsymbol{x}'(a) \in \mathcal{B}$. □

Corollary 1. *For the termination game over sound and weakly deterministic negotiations, the following holds:*

(a) the game collapses to a two-player game;
(b) memoryless strategies suffice for both players;
(c) the winner and the winning strategy can be computed in $O(|R| * |A|)$ time, where A is the set of agents and $|R|$ the total number of outcomes of the negotiation.

Proof. (a) and (b): The attractor computation and the strategies used in the proof of Theorem **??** are independent of the choices of Scheduler; the strategies are memoryless.
c) We describe an algorithm that computes the set $\mathcal{A}$ and the winning strategy.

```
array[][] count
array[] outcomes
array[] strategy
set border
set $\mathcal{A}$
for every atom $n$:
    outcomes[$n$] = |$R_n$|
    for every outcome $r \in R_n$:
        count[$n$][$r$] $\leftarrow$ |$P_n$|
$\mathcal{A} \leftarrow \{n_f\}$
border $\leftarrow \{n_f\}$
outcomes[$n_f$] $\leftarrow 0$
for every $r \in R_{n_f}$
    count[$n_f$][r] $\leftarrow 0$
while border != $\emptyset$
    choose and remove an atom $n$ from border
    for every atom $n' \in \mathcal{N}$ where $\mathcal{X}(n', a, r) = n$ for some $a \in P'_n, r \in R'_n$
        count[$n'$][$r$]−−
        if count[$n'$][$r$] = 0 and $n' \in \mathcal{N}_1$
            outcomes[$n'$] $\leftarrow 0$
            strategy[$n'$] $\leftarrow r$
            add $n'$ to border
        if count[$n'$][$r$] = 0 and $n' \in \mathcal{N}_2$
            outcomes[$n'$]−−;
            if outcomes[$n'$] = 0
                add $n'$ to border
return $\mathcal{A}$
```

Intuitively, *count*$[n][r]$ counts the ports of atom n that do not reach the attractor with outcome r, *outcomes*$[n]$ counts the outcomes of n for which *count*$[n][r] \neq 0$ or is zero of n is in the attractor.

This algorithm terminates and computes $\mathcal{A}$, the check $n_0 \in \mathcal{A}$ is done by checking whether *outcomes*$[n_0]$ is zero. Furthermore, the strategy for Player 1 is computed in *strategy*$[n]$. Every outcome is inspected at most once per agent, thus the running time is at most $O(|R| * |A|)$. This leads to a total running time of at most $O(|R| * |A| + |N|) \in O(|R| * |A|)$. Computing the strategy for Player 2 can be done in a straightforward loop over all outcomes choosing any outcome for which *count*$[n][r] \neq 0$ in running time $O(R)$. □

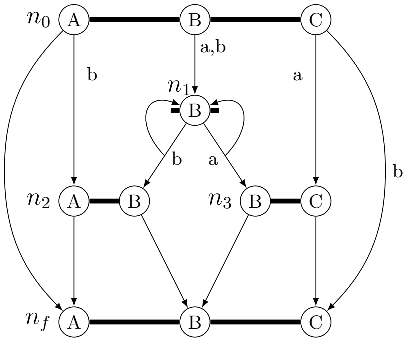

Fig. 5. A negotiation game where memoryless strategies do not suffice

We conclude this section by pointing out that memoryless strategies indeed only suffice for weakly deterministic sound negotiation games. Figure **??** shows a negotiation game, sound and nondeterministic. Imagine Agent B tries to enforce termination against the coalition of A and C, thus by majority Player 1 controls n_1, Player 2 controls the remaining atoms. This game requires a strategy with memory for Player 1 to win: Player 1 has to mirror Player 2's initial choice to win, otherwise an endless loop will be the result. A similar game that is deterministic but unsound can be obtained by removing the edges looping back to n_1. Again, the initial decision has to be mirrored, the result of deciding "wrong" will be a deadlock.

We still have to consider the possibility that requiring soundness alone, without the addition of weak determinism, already reduces the complexity of the termination problem. The following theorem shows that this is not the case, and concludes our study.

Theorem 6. *The termination problem is EXPTIME-hard for sound negotiation arenas.*

Proof. We modify the construction of Theorem **??** so that it yields a sound negotiation.

Recall that a negotiation is sound if every atom is fired in at least one occurrence sequence, and every occurrence sequence can be extended to a sequence that fires the final atom n_f. There are two reasons why the negotiation $\mathcal{N}_{M,x}$ defined in the proof of Theorem **??** is unsound in general:

– If M does not accept, then the final atom n_f is not reachable.
– If the head can never reach cell k with symbol α in it, then the atom $n_{q,\alpha,k}$ never occurs in any occurrence sequence,.

To solve these problems, we modify the construction of Theorem **??**. The negotiation of Theorem **??**, which we now call the old negotiation, just simulates a run of the machine. The new negotiation also does that in a first stage, but it also exhibits additional behavior. Intuitively, if the run rejects, then the new negotiation enters a "anything is possible"- mode, or *anything-mode* for short: the negotiation is then able to repeatedly fire any non-final atom, or fire the final atom and terminate. Furthermore, if the run accepts, then the new negotiation can either directly fire n_f, or enter the anything-mode.

Now we can argue why the new negotiation is sound. Since, by assumption, every run of the machine either accepts of rejects, the negotiation can always reach the anything-mode, which allows it to fire any atom, including the final atom.

However, we also have to guarantee that Player 1 has a winning strategy in the new negotiation iff the machine accepts. For this, we ensure two things. First, when the run accepts, the choice between firing n_f and entering the anything-mode is taken by Player 1. The winning strategy for Player 1 is to choose n_f. Second, in the anything-mode the choice whether to fire the final atom or some other atom is controlled by Player 2; therefore, if the anything-mode is entered after a reject run, Player 2 wins by never selecting the final atom.

We implement this idea by introducing a new agent S and two new atoms n_1, n_2. All agents, including S, participate in both n_1 and n_2, but n_0, n_1, n_2, n_f are the only atoms in which S participates. After n_0 occurs, agent S is ready to engage in both n_f and n_1. Firing n_1 signals the start of the anything-mode. After firing n_1, all agents move to n_2. Atom n_2 has two outcomes, s and f: after the outcome f all agents move to n_f, and so by choosing this outcome the negotiation terminates. After the outcome s, all agents but S are ready to engage in any atom, while S is only ready to engage in n_2 again. So this outcome corresponds to choosing any non-final atom of the old negotiation and firing it. But we still have to guarantee that after this atom is fired, atom n_2 is enabled again. For this, we add n_2 as n_2 to $\mathcal{X}(n, a, r)$ for every atom $n \neq \{n_1, n_2, n_f\}$, agent $a \neq S$, and outcome r

To ensure that the anything-mode is entered if the run of the Turing machine rejects, atoms of the form $n_{q_{reject},\alpha,k}$ are given one single outcome, and the $\mathcal{X}()$ function is modified as follows: P and I move to n_1, and we add n_1 to the set $\mathcal{X}(n, C_k, r)$ for every atom n and outcome r. So after $n_{q_{reject},\alpha,k}$ fires n_1 is the only atom enabled.

Now we ensure that if the run of the Turing machine accepts, then Player 1 can decide whether to terminate or enter the anything-mode. For this we let atoms of the form $n_{q_{accept},\alpha,k}$ to be controlled by Player 1, and add to them a second outcome that enables n_1 (additionally to the one enabling n_f).

The formal description of this construction can be found in the appendix.

6 The final-outcome problem

We demonstrate that the algorithm can also be used to solve the last-outcome problem. Going back to the negotiation of Father, Mother and two Daughters, imagine the goal of daughter D1 is to get a “yes” answer for herself, but a ‘no” answer for D2, who always spoils the fun. Will a coalition with one parent suffice to achieve the goal? To answer this question, we modify the negotiation before applying the construction as shown in Figure **??**. We introduce dummy atoms n_{yes} and n_{no} for each daughter, and we redirect“yes” edges leading to n_f to n_{yes}, analogously for n_{no}.

We apply the algorithm starting from all the dummy atoms representing edges we want to be taken. So we start from the set containing atom n_{yes} of D1 and n_{no} of D2. Applying the algorithm for the coalition Father-D1 yields $\{n_{yes1}, n_{no2}, n_4\}$ as attractor; For Mother-D1 we get $\{n_{yes1}, n_{no2}, n_6\}$. So neither parent has enough influence to achieve the desired outcome.

7 Conclusions and related work

We have started the study of games in the negotiation model introduced in [**?**,**?**]. Our results confirm the low computational complexity of deterministic

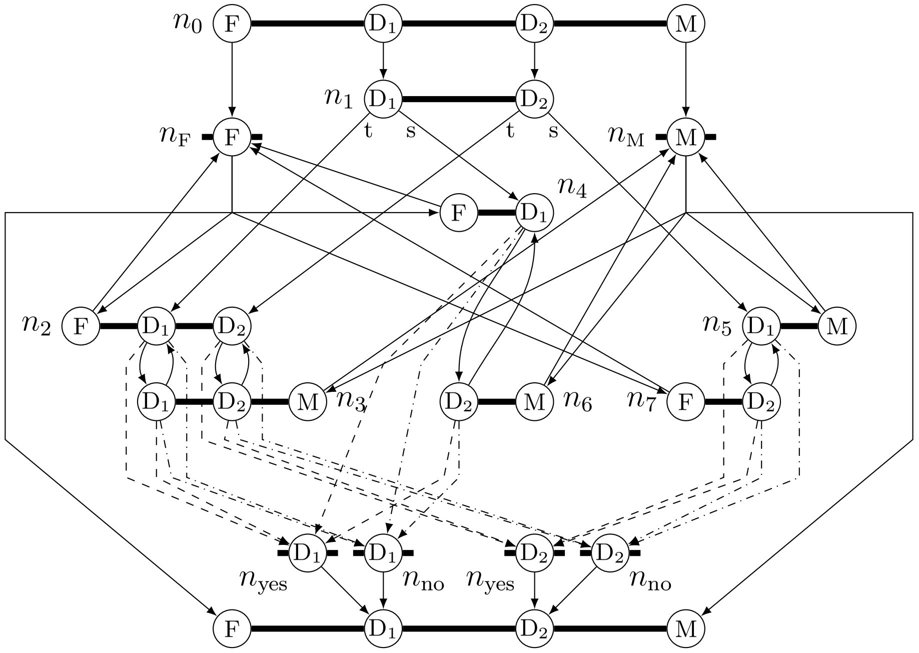

Fig. 6. Applying the algorithm to the final-outcome problem

negotiations, however with an important twist: while even the simplest games are EXPTIME-hard for arbitrary deterministic negotiations, they become polynomial in the *sound* case. So soundness, a necessary feature of a well designed negotiation, also turns out to have a drastic beneficial effect on the complexity of the games.

We have shown that our games are also polynomial for sound and weakly deterministic negotiations. However, the complexity of deciding soundness for this case is unknown. We conjecture that it is also polynomial, as for the deterministic case.

We have only considered 2-player games, since in our settings the behavior of the third player (the Scheduler) is either irrelevant, or is controlled by one of the other two players. We intend to study the extension to a proper 3-player game, or to a multi player game, in particular for negotiations where the game objective involves the internal state of the agents.

8 Appendix

8.1 Proofs of section ??

Theorem ??. The termination problem is EXPTIME-hard even for deterministic negotiations in which every reachable marking enables at most one atom.

Proof. We now modify the initial construction to yield a deterministic (but still unsound) negotiation. First we present a general construction to remove nondeterminism from arbitrary negotiations, we then apply this construction with some alterations to fit the needs of negotiation games. The transformation is as follows: for any triple (n, a, r) such that $\mathcal{X}(n, a, r) = \{n_1, \ldots, n_k\}$ for some $k > 1$, we introduce a new atom $n_{a,r}$, assigned to Player 1, with a as single party, and with k different outcomes $r_1, \ldots, r_k$, and redefine $\mathcal{X}$ as follows: $\mathcal{X}(n, a, r) = \{n_{a,r}\}$, and $\mathcal{X}(n_{a,r}, r_i\} = \{n_i\}$ for every $1 \leq i \leq k$. Intuitively, in the original negotiation after the outcome $(n_0, \mathtt{st})$ agent a is ready to engage in $n_1, \ldots, n_k$; in the new negotiation, after $(n_0, \mathtt{st})$ a commits to *exactly one* of $n_1, \ldots, n_k$, and the choice is controlled by Player 1. Unfortunately, we cannot apply this construction directly because of the following problem: An agent C_k has to decide in which of the $n_{q,\alpha,k}$ they want to participate next, therefore he needs to know the state q in which the head will arrive next in cell k. But this may not be decided until the head arrives in cell k. Therefore we need to postpone this decision until the head actually arrives in cell k.

This is achieved by the following alterations: In the new construction, after firing the outcome of $n_{q,\alpha,k}$ for transition $(q', \beta, R) \in \delta(q, \alpha)$, agent C_k moves to $n_{\beta,k}$, and agents I and P move to $n_{q',k+1}$. Intuitively, C_k waits for the head to return to cell k, while agents I and P proceed. Atom $n_{q',k+1}$ has an outcome for every tape symbol γ. Intuitively, at this atom Player 1 guesses the current tape symbol in cell $k + 1$; the winning strategy corresponds to guessing right. After guessing, say, symbol γ, agent I moves directly to $n_{q',\gamma,k+1}$, while P moves to

$n_{\gamma,k+1}$. Atom $n_{\gamma,k+1}$ has one outcome for every state of M. Intuitively, Player 1 now guesses the current control state q', after which both P and C_{k+1} move to $n_{q',\gamma,k+1}$. Notice that all moves are now deterministic.

If Player 1 follows the winning strategy, then the plays mimic those of the old construction: a step like $(n_{q,\alpha,k}, (q', \beta, R))$ in the old construction, played when the current symbol in cell $k+1$ is γ, is mimicked by a sequence $(n_{q,\alpha,k}, (q', \beta, R))$ $(n_{q',k+1}, \gamma)$ $(n_{\gamma,k+1}, q')$ of moves in the new construction.

Formally, the construction is the following:

- The set of agents is

$$A = \{I, P\} \cup \{C_k : k \in \{0, ..., n-1\}\}$$

- The set of atoms is

$$\begin{aligned} N = \{n_0, n_f\} &\cup \{n_{q,\alpha,k} : (q, \alpha, k) \in Q \times \mathit{\Gamma} \times \{0, ..., n-1\}\} \\ &\cup \{n_{\alpha,k} : (\alpha, k) \in \mathit{\Gamma} \times \{0, ..., n-1\}\} \\ &\cup \{n_{q,k} : (q, k) \in Q \times \{0, ..., n-1\}\} \end{aligned}$$

- The set N_1 of atoms owned by Player 1 contains n_0, n_f, all atoms $n_{\alpha,k}$, all atoms $n_{q,k}$, and all atoms $n_{q,\alpha,k}$ such that $g(q) \neq \wedge$. The set N_2 contains all other atoms.
- The set of parties of each atom is given by

$$P_{n_0} = P_{n_f} = A$$

$$P_{n_{q,\alpha,k}} = \{I, P, C_k\} \text{ for every } n_{q,\alpha,k} \in N$$

$$P_{n_{\alpha,k}} = \{P, C_k\} \text{ for every } n_{\alpha,k} \in N$$

$$P_{n_{q,k}} = \{I, P\} \text{ for every } n_{q,k} \in N$$

- The set of outcomes of each atom is defined as follows: $R_{n_0} = \{\text{st}\}$, $R_{n_f} = \{\text{end}\}$, and

$$R_{n_{q,\alpha,k}} = \begin{cases} \{\text{f}\} & q = q_{accept} \\ \{\text{s}\} & q = q_{reject} \\ \{r_\tau : \tau \in \delta(q, \alpha)\} & \mathit{otherwise} \end{cases} \quad \text{for every } n_{q,\alpha,k} \in N$$

$$R_{n_{q,k}} = \{r_\alpha : \alpha \in \mathit{\Gamma}\} \text{ for every } n_{q,k} \in N$$

$$R_{n_{\alpha,k}} = \{r_q : q \in Q\} \text{ for every } n_{\alpha,k} \in N$$

- The $\mathcal{X}(\_, P, \_)$ function for agent P is given by:

$$\mathcal{X}(n_0, P, \text{st}) = \{n_{\alpha,0} : \alpha \text{ is the initial content of cell } 0\}$$

$$\mathcal{X}(n_{q,\alpha,k}, P, r_\tau) = \begin{cases} \{n_{q',k-1}\} & \text{if } \tau = (q', \_, L) \\ \{n_{q',k+1}\} & \text{if } \tau = (q', \_, R) \end{cases}$$

$$\mathcal{X}(n_{q,k}, P, r_\alpha) = \{n_{\alpha,k}\}$$

$$\mathcal{X}(n_{\alpha,k}, P, r_q) = \{n_{q,\alpha,k}\}$$

$$\mathcal{X}(n_{q_{accept},\alpha,k}, P, \mathrm{f}) = \{n_f\}$$

$$\mathcal{X}(n_{q_{reject},\alpha,k}, P, \mathrm{s}) = \{n_{q_{reject},\alpha,k}\}$$

- The $\mathcal{X}(\_, I, \_)$ function for agent I is given by:

$$\mathcal{X}(n_0, I, \mathrm{st}) = \{n_{q_0,\alpha,0} : \alpha \text{ is the initial content of cell 0}\}$$

$$\mathcal{X}(n_{q,\alpha,k}, I, r_\tau) = \begin{cases} \{n_{q',k-1}\} & \text{if } \tau = (q', \_, L) \\ \{n_{q',k+1}\} & \text{if } \tau = (q', \_, R) \end{cases}$$

$$\mathcal{X}(n_{q,k}, I, r_\alpha) = \{n_{q,\alpha,k}\}$$

$$\mathcal{X}(n_{q_{accept},\alpha,k}, I, \mathrm{f}) = \{n_f\}$$

$$\mathcal{X}(n_{q_{reject},\alpha,k}, I, \mathrm{s}) = \{n_{q_{reject},\alpha,k}\}$$

- Finally, the $\mathcal{X}(\_, C_k, \_)$ function for agent C_k is given by

$$\mathcal{X}(n_0, C_k, \mathrm{st}) = \{n_{\alpha,k} : \alpha \text{ is the initial content of cell k}\}$$

$$\mathcal{X}(n_{q,\alpha,k}, C_k, r_\tau) = \{n_{\beta,k}\} \text{ if } \tau = (\_, \beta, \_)$$

$$\mathcal{X}(n_{\alpha,k}, C_k, r_q) = \{n_{q,\alpha,k}\}$$

$$\mathcal{X}(n_{q_{accept},\alpha,k}, C_k, \mathrm{f}) = \{n_f\}$$

$$\mathcal{X}(n_{q_{reject},\alpha,k}, C_k, \mathrm{s}) = \{n_{q_{reject},\alpha,k}\}$$

□

8.2 Proofs of section ??

Theorem ??. The termination problem is EXPTIME-hard for sound negotiation arenas.

Proof. Recall that a negotiation is sound if every atom is fired in at least one occurrence sequence, and every occurrence sequence can be extended to a sequence that fires the final atom n_f. There are two reasons why the negotiation $\mathcal{N}_{M,x}$ defined in the proof of Theorem **??** is unsound in general:

- If M does not accept, then the final atom n_f is not reachable.
- If the head can never reach cell k with symbol α in it, then the atom $n_{q,\alpha,k}$ never occurs in any occurrence sequence,.

To solve these problems, we modify the construction of Theorem **??**. The negotiation of Theorem **??**, which we now call the old negotiation, just simulates a run of the machine. The new negotiation also does that in a first stage, but it also exhibits additional behavior. Intuitively, if the run rejects, then the new negotiation enters a “anything is possible”- mode, or *anything-mode* for short: the negotiation is then able to repeatedly fire any non-final atom, or fire the final atom and terminate. Furthermore, if the run accepts, then the new negotiation can either directly fire n_f, or enter the anything-mode.

The visual representation of the construction (with a simplified representation of the TM) can be found in Figure **??**. Dashed edges always mean that all agents C_k have nondeterministic edges for every outcome to that port/from that port to all other ports.

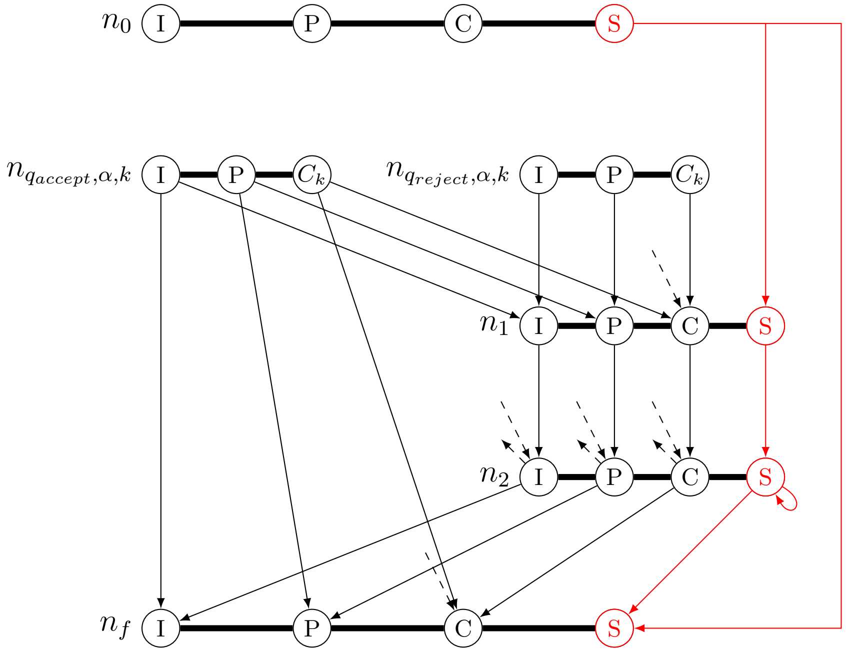

Fig. 7. Constructing a sound negotiation for the TM reduction. C means all agents C_k

Now we can argue why the new negotiation is sound. Since, by assumption, every run of the machine either accepts of rejects, the negotiation can always reach the anything-mode, which allows it to fire any atom, including the final atom.

However, we also have to guarantee that Player 1 has a winning strategy in the new negotiation iff the machine accepts. For this, we ensure two things. First, when the run accepts, the choice between firing n_f and entering the anything-mode is taken by Player 1. The winning strategy for Player 1 is to choose n_f.

Second, in the anything-mode the choice whether to fire the final atom or some other atom is controlled by Player 2; therefore, if the anything-mode is entered after a reject run, Player 2 wins by never selecting the final atom.

We implement this idea by introducing a new agent S and two new atoms n_1, n_2. All agents, including S, participate in both n_1 and n_2, but n_0, n_1, n_2, n_f are the only atoms in which S participates. After n_0 occurs, agent S is ready to engage in both n_f and n_1. Firing n_1 signals the start of the anything-mode. After firing n_1, all agents move to n_2. Atom n_2 has two outcomes, s and f: after the outcome f all agents move to n_f, and so by choosing this outcome the negotiation terminates. After the outcome s, all agents but S are ready to engage in any atom, while S is only ready to engage in n_2 again. So this outcome corresponds to choosing any non-final atom of the old negotiation and firing it. But we still have to guarantee that after this atom is fired, atom n_2 is enabled again. For this, we add n_2 as n_2 to $\mathcal{X}(n, a, r)$ for every atom $n \neq \{n_1, n_2, n_f\}$, agent $a \neq S$, and outcome r

To ensure that the anything-mode is entered if the run of the Turing machine rejects, atoms of the form $n_{q_{reject},\alpha,k}$ are given one single outcome, and the $\mathcal{X}()$ function is modified as follows: P and I move to n_1, and we add n_1 to the set $\mathcal{X}(n, C_k, r)$ for every atom n and outcome r). So after $n_{q_{reject},\alpha,k}$ fires n_1 is the only atom enabled.

Now we ensure that if the run of the Turing machine accepts, then Player 1 can decide whether to terminate or enter the anything-mode. For this we let atoms of the form $n_{q_{accept},\alpha,k}$ to be controlled by Player 1, and add to them a second outcome that enables n_1 (additionally to the one enabling n_f).

Formally, here are the changes to the construction of Theorem **??**:

- S is added to the set of agents: $A' = A \cup \{S\}$
- n_1, n_2, controlled by Player 2 are added to the atoms: $N' = N \cup \{n_1, n_2\}$ and $N_2' = N_2 \cup \{n_1, n_2\}$
- All agents participate in n_1, n_2 (and S in n_0, n_f):

$$P'_{n_0} = P'_{n_f} = P'_{n_1} = P'_{n_2} = A'$$

- The outcomes for n_1 and n_2 are define by: $R'_{n_1} = \{\text{s}\}$, $R'_{n_2} = \{\text{s,f}\}$
- All atoms $n_{q_{accept},\alpha,k}$ get a second outcome: $R'_{n_{q_{accept},\alpha,k}} = \{\text{s,f}\}$
- The $\mathcal{X}(n_1, \_, \_)$ function for atom n_1 is given by

$$\mathcal{X}'(n_1, a, s) = \{n_2\} \text{ for all } a \in P_{n_1}$$

- The $\mathcal{X}(n_2, \_, \_)$ function for atom n_2 is given by

$$\mathcal{X}'(n_2, a, \text{s}) = \{n \in N' \backslash \{n_f\} : a \in P_n\} \text{ for all } a \in P_{n_2} \backslash \{S\}$$

$$\mathcal{X}'(n_2, a, \text{f}) = \{n_f\} \text{ for all } a \in P_{n_2}$$

- The $\mathcal{X}(\_, S, \_)$ function for agent S is given by:

$$\mathcal{X}'(n_0, S, \text{st}) = \{n_f, n_1\}$$

$$\mathcal{X}'(n_2, S, \mathrm{s}) = \{n_2\}$$
$$\mathcal{X}'(n_2, S, \mathrm{f}) = \{n_2\}$$

- The $\mathcal{X}(n, \_, \_)$ function for all accepting and rejecting atoms $n_{q_{accept}, \alpha, k}$, $n_{q_{reject}, \alpha, k}$ is modified to:

$$\mathcal{X}'(n_{q_{accept}, \alpha, k}, a, \mathrm{s}) = \{n_1\} \text{ for all } a \in P'_{n_{q_{accept}, \alpha, k}}$$
$$\mathcal{X}'(n_{q_{accept}, \alpha, k}, a, \mathrm{f}) = \{n_f\} \text{ for all } a \in P'_{n_{q_{accept}, \alpha, k}}$$
$$\mathcal{X}'(n_{q_{reject}, \alpha, k}, a, \mathrm{s}) = \{n_1\} \text{ for all } a \in P'_{n_{q_{reject}, \alpha, k}}$$

- And finally, all agents except S are ready to take part in n_2 at any time:

$$\mathcal{X}'(n, a, r) = \mathcal{X}'(n, a, r) \cup \{n_2\} \text{ for all } n \in n \in N' \backslash \{n_f\}, a \in P'_n \backslash \{S\}, r \in R'_n$$

Now all atoms are enabled after some sequence, the end atom can always be reached, and P_1 can enforce the end atom iff M terminates in q_{accept}. □